# Developing a Physics Problem Simulation Engine
# TJHSST Senior Research Project Proposal
# Computer Systems Lab 2009-2010

Neal Milstein

October 28, 2009

**Abstract**

This project will attempt to create an accurate physics simulation engine that can be used by physics students to visualize and solve physics textbook problems. The project can be broken down into two parts: the physics engine, and the simulation interface, both of which will be written in Java. The interface will allow the user to draw various physical objects, such as masses and springs, and then run the simulation to see how they interact over a time interval. The engine will utilize modern mathematical techniques, such as Runge-Kutta integration, to properly simulate physical simulations. Following the engine's completion, a variety of physics problems will be simulated, and the accuracy of the engine will be assessed.

**Keywords:** physics engine, graphic simulation, mathematical techniques

## 1 Introduction

The proper visualization of physical situations can be a difficult process for physics students, especially beginners. It is not always possible to simulate situations using actual physical components in the classroom, and computers can therefore be a very valuable tool towards helping students visualize physics problems. Furthermore, when physics problems can be visualized using a computer, specific interactions can be studied over and over again at varying speeds and perspectives, something that cannot be easily done in a classroom. The goal of this project is to develop such an accurate physics problem simulation engine, so that physical setups can be visualized on a computer, and the complex interactions between various objects can be visualized.

## 2 Background

One of the primary challenges to this project will be an accurate collision detection system. A number of mathematical and computer science techniques have been developed to efficiently detect such collisions. The separating axis theorem, for example, states that if two non-concave bodieas are not intersecting, a line can be drawn between them that does not intersect either body. This theorem will be used in the project's collision detection system, along with optimization techniques such as a sweep and prune model.

Another aspect of collision detection that must be worked out is whether to use *a posteriori* or *a priori* to detect collisions. The two methods are summarized below:

1. *a posteriori* Under this method of collision detection, objects are first advanced in the simulation (i.e. moved or modified), and then the physics engine checks for collisions. This is a simple method of detecting collisions, as it requires very few advance calculations to detect if the collisions took place. However, it is then more difficult to determine how the objects should react to the collisions.

2. *a priori* With this method, the trajectories of objects are accurately calculated, and the engine then detects collisions. This makes the collision detection significantly harder to develop, the reactions of objects to collisions are much easier to model.

Although the first step of the project will be to model kinematics, more advanced techniques may be modeled in the future. Research has already been completed on physics simulation techniques developed by other software groups. One group, for example, opted to use surface interaction detection, through which human hands, when properly interfaced with a computer, could control a variety of physically simulated objects. This approach hid many of the preprogrammed aspects of the physics engine, and it allowed for the utilization of the physics engine in application without implementing complex physical object interactions.

# 3 Development

Project development will first entail a simple graphics driver, which will allow for the physical components in the project to be simulated. This will allow for initial testing and evaluation of results, as it will be hard to initially determine the success of the simulation engine from simple console displays. Then, the physics simulation engine will be developed. Initially, the framework for the engine will be planned out, and the method by which object interactions are calculated will be worked out. Following this planning stage, simple kinematics will be implemented in the engine, allowing for the simulation of many beginner's physics problems. Collision detection will also be implemented at this stage, to allow objects within the simulation to interact. Following these steps, more advanced phenomena, such as linear momentum and gravity will be simulated. The entire simulation engine will be coded in Java.

The interface will be developed following the completion of the major elements of the physics engine. The interface will allow users to draw objects and components on the screen, upon which the simulation can be run, and the objects will interact with one another. The interface will resemble that of typical applications, allowing for tools to be selected to draw objects and a pane for viewing the properties of objects.

Following this development, the simulation engine will be tested in full to assess the accuracy of the results. The simulation engine will be functioning correctly if it yields result close to those achieved by hand on paper. The interface will be correctly implemented if it allows the user to properly draw physical components and view the properties of these components.

# 4 Expected Results

The physics simulation will be expected to yield accurate data as to the state of a physical setup over a given time interval. The results obtained from the engine will be compared to those calculated by hand. Furthermore, the interface must be able to properly visualize modeled physics problems, and it is expected to be able to accurately display visual models of the physics setup.

# References


[1] O. Hilliges, S. Izadi, D. Kirk, A. Garcia-Mendoza, A. D. Wilson, "Bringing Physics to the Surface", http://portal.acm.org/citation.cfm?id=1449715.1449728

[2] C. B. Price, "The usability of a commercial game physics engine to develop physics educational materials: An investigation", http://portal.acm.org/citation.cfm?id=1401790.1401794